

Final Project Report

Dr. Jeremy Daily

SYSE-548

Fall 2025 Semester

Spencer Beer

beersc@colostate.edu

Colorado State University

December 4, 2025

Marine IoT Inherits Classic Consumer-IoT Security Gaps: Utilizing SysML to Realize Threats
and Mitigations in a Proprietary System

Submitted as the final report for the course project.

Contents

1	Introduction	2
1.1	Documentation and Iterative Refinement with SysML	2
1.1.1	MBSE: Model Based Systems Engineering	2
1.1.2	SysML: Systems Modeling Language	2
1.1.3	Reverse Engineering	2
1.1.4	RSysML: Reverse SysML	3
1.1.5	CVMS: Cybersecurity Vulnerability Management System	3
1.2	Background and Objective	4
1.2.1	National Marine Electronics Association	4
1.2.2	Marine Gateways	5
1.3	YDWG-02: The Wireless Gateway to Remote Control of Marine Systems	6
1.3.1	Reconnaissance and Open Source Intelligence	7
2	Early Model Elicitation	7
3	Model Refinement with RSysML	8
3.1	Structure	8
3.2	Behavior	8
3.3	Threats	8
3.4	Mitigations	8
3.4.1	Implementation	9
3.4.2	Testing	10
3.4.3	Vulnerabilities	10
4	Reflection	10
A	Supporting SysML Figures	13
B	Supporting Code	26

1 Introduction

1.1 Documentation and Iterative Refinement with SysML

1.1.1 MBSE: Model Based Systems Engineering

Model-Based Systems Engineering replaces traditional document-driven processes with structured, formal models that capture system design, requirements, behavior, and verification artifacts across the development lifecycle.

1.1.2 SysML: Systems Modeling Language

SysML was standardized as a Unified Modeling Language (UML) profile for systems engineering [1]. It supports MBSE by offering a standardized set of diagrams categorized as:

- Structural: Block Definition Diagrams (BDDs) for component hierarchies and part definitions; Internal Block Diagrams (IBDs) for internal connectivity and interfaces
- Behavioral: Activity Diagrams (ACTs) to model workflows and logic; Sequence Diagrams (SEQs) for time-ordered interactions; State Machine Diagrams (STMs) for state-dependent behavior; and Use Case Diagrams (UCs) for high-level functional goals
- Requirement: Requirement (REQ) diagrams define and trace functional, performance, and security needs

1.1.3 Reverse Engineering

Reverse engineering is commonly performed on black-box systems where internal structure and source code are unavailable. Analysts must rely on observable behavior, inputs, outputs, and performance characteristics, to infer internal logic. Without access to code or instrumentation, testers use *black-box testing* to probe system responses through lightweight observation, such as monitoring throttle, brake, or speed signals to study an uninstrumented automotive controller.

Recent work shows that internal modes can be inferred directly from telemetry. Ataiefard *et al.* use a hybrid neural network to identify latent state “phases” in UAV autopilot logs, achieving over 90% accuracy in classifying operational modes from multivariate time-series data [2]. This demonstrates how behavioral models can be reconstructed even without source code.

Traditional RE techniques remain as well. Static analysis inspects binaries with disassemblers (e.g., Ghidra [3]) to recover control flow, call graphs, and data structures. Dynamic analysis uses debuggers, sandboxing, tracing, and fault injection to study runtime behavior. Network analysis and protocol fuzzing help uncover message formats and undocumented behaviors in distributed or IoT systems. For embedded Cyber-Physical Systems, engineers may probe hardware buses, extract firmware , or analyze flash memory. When firmware extraction is impossible, timing, side-channel signals, and system identification methods still reveal internal dynamics. Together, these techniques enable reconstruction of system structure and behavior, critical for verifying whether a system’s actual operation aligns with inferred understanding of a system.

1.1.4 RSysML: Reverse SysML

Utilizing systems engineering best practices, such as the V model, the concept of utilizing the systems modeling language for reconstructing legacy, proprietary, or otherwise closed-source systems helps organize the documentation and iteratively refine new systems. While prior reverse-engineering efforts have employed SysML primarily as a presentation medium, to visualize extracted architectures, interface diagrams, or partial behavioral fragments, few have treated SysML as a reconstruction framework in its own right. Existing studies demonstrate that SysML can depict reverse-engineered system elements, yet none explicitly leverage it as an iterative modeling environment that integrates new architectural and behavioral insights over successive refinement cycles [4, 5, 6]. The approach integrates the traceability and structure of building a wholistic model that eventually traces back to the requirements (i.e., root of structure or behavior).

1.1.5 CVMS: Cybersecurity Vulnerability Management System

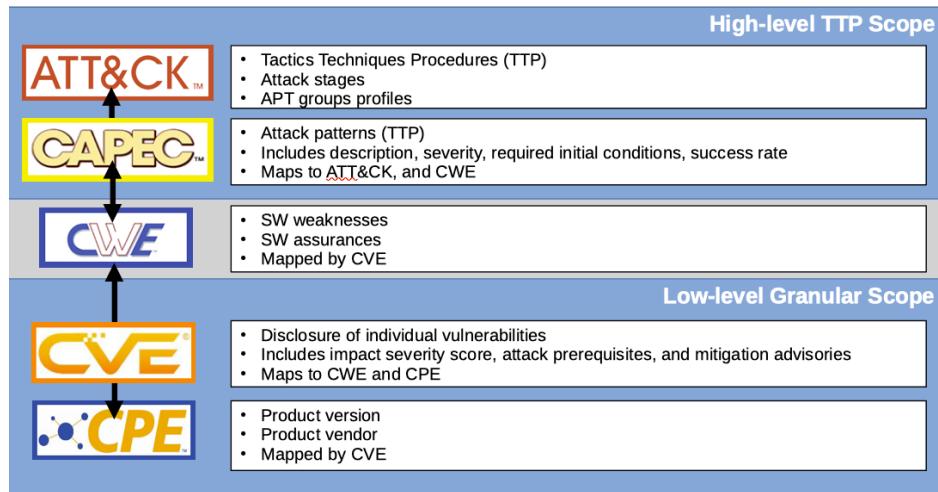


Figure 1.1: MITRE Framework [7]

MITRE Corporation is a U.S. not-for-profit that operates several federally funded research and development centers supporting government agencies. It has been a central contributor to cybersecurity for over two decades. MITRE launched the Common Vulnerabilities and Exposures (CVE) program in 1999 and continues to maintain it with government and industry partners [8]. It later introduced the Common Weakness Enumeration (CWE) in the early 2000s and the Common Attack Pattern Enumeration and Classification (CAPEC) catalog around 2007, solidifying its role in the vulnerability management ecosystem.

Frameworks such as CVE, CWE, and CAPEC provide complementary perspectives on vulnerabilities and risk:

- CVE assigns unique identifiers to publicly disclosed vulnerabilities, enabling a universal reference system for tracking and correlating issues across organizations
- CWE categorizes underlying software and hardware weaknesses, supporting root-cause analysis, tool coverage assessment, and mitigation planning
- CAPEC catalogs adversary attack patterns, describing how weaknesses are exploited and

helping practitioners prioritize defenses and model attack behavior

Used together, these frameworks create traceability and structure from a specific vulnerability (CVE), to its underlying weakness (CWE), to the attack patterns (CAPEC) that typically exploit it. This integrated taxonomy improves risk modeling, vulnerability prioritization, and mitigation strategy development.

In a cybersecurity vulnerability management system (CVMS), “linking to real-world evidence” means mapping empirical findings, such as reverse-engineering artifacts, into these taxonomies. A modeling language like SysML can express these relationships, reveal risk propagation paths, and support design decisions. SysML diagrams can show, for example, how a weakness contributes to a vulnerability and how that vulnerability aligns with an attack pattern, as well as which mitigations break the causal chain.

The CVMS SysML profile defines stereotypes that formalize how cybersecurity concepts attach to system elements:

- Threaten: a weakness, vulnerability, or attack pattern demonstrably threatens a specific system element.
- Mitigate: a system element provides a verified reduction of risk for a given weakness, vulnerability, or attack pattern.
- PotentialThreat: a plausible but unconfirmed threat relationship exists.
- PotentialMitigation: a plausible but unvalidated mitigation relationship exists.
- Weakness, Vulnerability, and AttackPattern: library elements representing CWE, CVE, and CAPEC concepts, respectively:
 - Weakness (CWE) maps to SysML requirements as root causes.
 - Vulnerability (CVE) maps to structural components as concrete system-specific issues.
 - AttackPattern (CAPEC) maps to behavioral elements as exploitation mechanisms.

This profile encodes the causal chain *Weakness* → *Vulnerability* → *Attack Pattern*, enabling consistent reasoning and end-to-end traceability from root causes to exploit behavior. The ability to express both confirmed and potential relationships supports iterative, model-based cybersecurity analysis within the CVMS. Through SysML relationship types such as trace, refine, deriveReqt, satisfy, and verify, enabling semantic linkage across abstraction layers assisting in risk analysis. The case study in this paper showcases how SysML and reverse engineering combine to realize threats and mitigations in marine gateways.

1.2 Background and Objective

1.2.1 National Marine Electronics Association

The National Marine Electronics Association (NMEA) has developed standards for the communication subsystems of boats, including:

1. A legacy serial data standard (RS-422/RS-232) for one-talker, many-listener systems, using ASCII ”sentences” (e.g., GPS, AIS, autopilots) [9]
2. A CAN-bus-based marine network standard (multi-talker/listener) using binary messages (PGNs) for instrumentation, engines, navigation systems [10]

3. An Ethernet/IPv6-based next-generation standard designed for high-bandwidth marine systems (video, radar, large sensor networks), compatible with earlier NMEA standards via gateways [11]

1.2.2 Marine Gateways

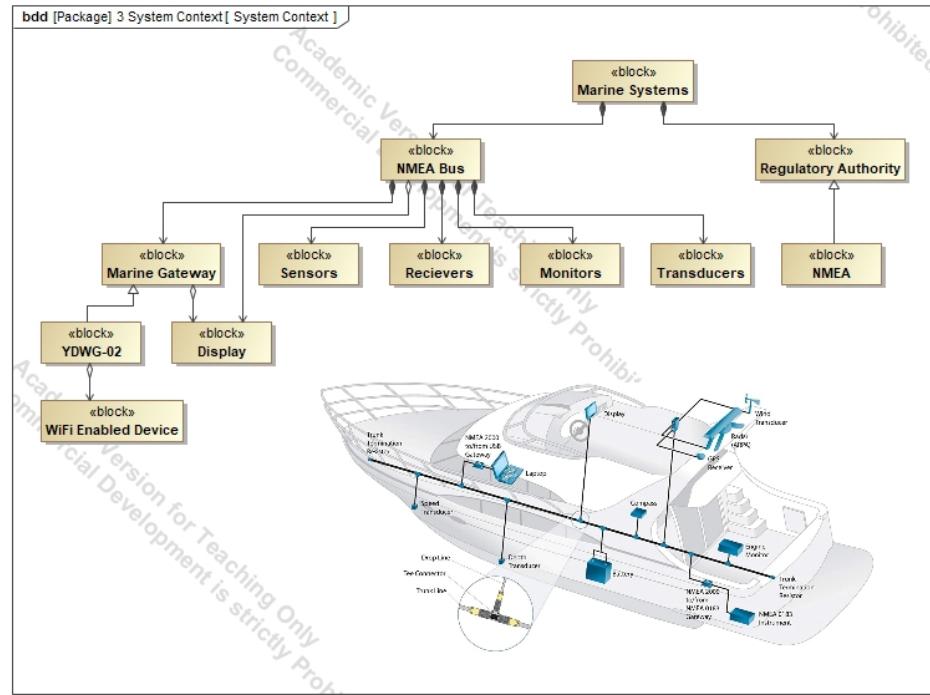


Figure 1.2: System Context of Marine Gateway [12]

In the marine vessel systems context, a gateway is a device (or module) that lets two (or more) different data networks or protocols talk to one another onboard a boat or ship. It “translates” or bridges between legacy and modern systems so data from sensors, instruments, engines, etc., can be integrated and shared. For example, a legacy instrument using one protocol might need to feed data into a network that uses a newer standard.

Such gateways are increasingly important because marine vessels often have a mix of equipment (navigation, engine, sensors, legacy/old and new) and standards evolve over time [9].

A few examples on the market at the time of this paper include:

1. The NGX-1 from Actisense which features conversions from NMEA 2000, 0183 and a PC interface [13]
2. The YDWN-02 WiFi Gateway that features NMEA 0183 to a PC interface [14]
3. The Maretron J2K100 which includes feature support for converting medium-to-heavy-duty protocols such as SAE J1939 to NMEA 2000 [15]

With the background on NMEA standards and the role of marine gateways established, the objective of this paper is to reconstruct and verify a marine gateway’s structure, behavior, and security properties using the Systems Modeling Language, along with other systems engineering practices (see Figure 1.3). Because vessels often mix older and newer systems, and the standards continue

to evolve, this work also aims to clarify how gateways function today and provide a starting point for developing improved marine Internet-of-Things (IoT) systems. Studies examining shipboard wireless networks have demonstrated that poorly configured or unsegmented wireless gateways can be exploited through rogue access points, man-in-the-middle attacks, and interception of crew or onboard device traffic. These findings highlight that vulnerabilities often arise not from exotic zero-day flaws but from weak authentication, default credentials, outdated firmware, and insecure bridging between IT and OT domains [16]. However, these studies fail to highlight is a *real-world* cyber-physical gateway, from the IT domain to the OT domain. This study increases the confidence in these findings by applying *real-world* threat modeling and mitigation approaches.

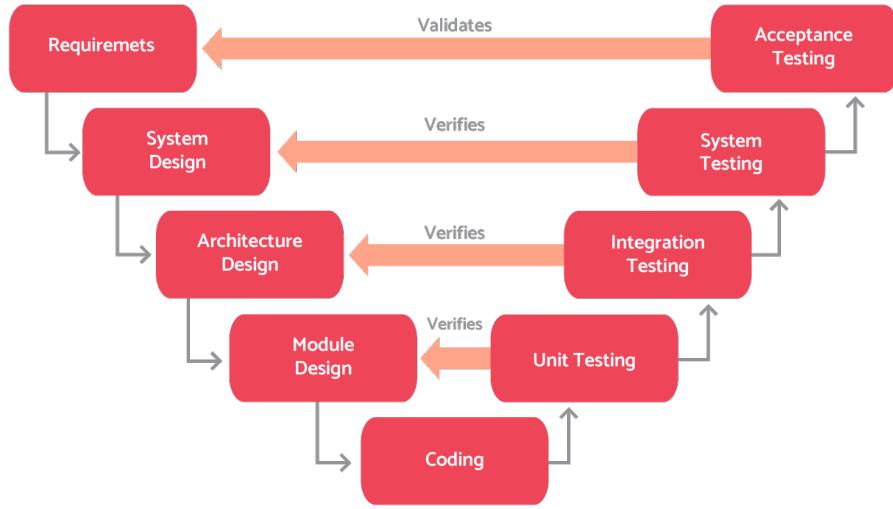


Figure 1.3: Systems Engineering V Model [17]

1.3 YDWG-02: The Wireless Gateway to Remote Control of Marine Systems

The system of interest for this case study is a marine wireless gateway: a thumb-sized device that bridges NMEA 2000, NMEA 0183, and RAW protocol streams over 2.4 GHz Wi-Fi. It enables bidirectional data flow between onboard sensors and tablets, supporting applications such as OpenCPN, Navionics, and Signal K. The gateway can also interface with Raymarine SeaTalk NG autopilot computers, offering both data visualization and control. This gateway is the YDWG-02 [18].

The manufacturer reported rapid growth since its founding in 2014. By late 2017, the company noted a 500% increase in sales over 2016. In 2018 they announced they had sold 3x more units than in 2017 and expanded to a network of about 40 dealers worldwide. Even during the pandemic and semiconductor shortages, demand grew: sales in 2021 were 26% higher than 2020 and 44% higher than 2019. Although exact unit counts were not disclosed, these growth rates imply a substantial user base. If sales were in the low hundreds in 2016, a 5x jump by 2017 and 3x by 2018 would put annual units in the thousands by 2018. With continued double-digit growth through 2021, it is reasonable to estimate total YDWG-02 devices in circulation in the high thousands (possibly approaching the low tens of thousands) globally by the mid-2020s, though exact figures remain proprietary. The company's year-end reports also mention OEM projects and use by charter or

fleet companies, implying these devices are not only DIY aftermarket gadgets but are finding their way into professional deployments on commercial or rental vessels as well [19].

Its operational domain includes small to medium recreational and commercial vessels, many of which integrate this device for navigation, telemetry, and cloud-based tracking. Despite its growing market share since 2014, several legacy security and documentation shortcomings persist, particularly the use of default passwords, the absence of encryption in transit, and broken access control. Its relevance lies in that it is both an IoT node, and a safety-critical control point. It presents an accessible and verifiable test platform in that it uses off-the-shelf components and provides observable network behaviors through WiFi and CAN-based marine buses.

1.3.1 Reconnaissance and Open Source Intelligence

Information was gathered from multiple open and technical resources. To start, the device included a vendor-supplied user manual published on the manufacturer website, which documents device operation, Wi-Fi defaults, server ports, RAW protocol format, and the firmware update procedure (WUPDATE.BIN). The manual's Appendix E explicitly describes the RAW message format used to convey NMEA-2000 frames over TCP/UDP, and several sections give the default network configuration (SSID "YDWG", default password "12345678", web UI at <http://192.168.4.1>, and default NMEA server port 1456) which were essential to reproducing normal operation and test setups [18].

Beyond the manual, community forums, developer notes, and open-source tooling documentation provided practical guidance for firmware extraction and protocol decoding; these sources also surfaced real-world use cases for the gateway and common misconfiguration patterns [20, 21]. Internet footprinting (Shodan) queries which located instances of the gateway accessible on the public Internet, confirming that factory configuration and exposed services are not purely theoretical attack vectors [22].

Part of the reconnaissance phase includes understanding (as closely as possible) normal operation. Given the manufacturer provided the manual for the device, it was straightforward to deduce the expected default network and protocol behaviors (Access Point mode vs Client mode, server port assignments, RAW/NMEA mapping rules, and logging/diagnostics endpoints). Those documented behaviors informed the first SysML structural hypotheses and guided where to attach measurement probes (TCP ports 1456/1458, http, OTA update process) during the active reverse-engineering stages.

2 Early Model Elicitation

The first stages of systems engineering, includes understanding the stakeholder needs. For the YDWG-02, this was the need for real-time data access, connectivity, and remote monitoring support (see Appendix A.1). Utilizing the reconnaissance earlier, system requirements were hypothesized, shown in Appendix A.2. For example, given the manual describes, "To access the Device web interface, connect to this Wi-Fi network and enter <http://192.168.4.1> in a web browser," then there must be a web-based user interface. This is later integrated into the user interface system requirement (SR6), as hypothesized, and derived from the needs described in Appendix A.1. After developing those initial requirements, structural and behavioral assumptions were then developed to solidify an inferred understanding of the marine gateway (see Appendix A.3 and A.4).

3 Model Refinement with RSysML

After gathering information online to build an initial SysML model, reverse engineering helps to reveal structural and behavioral relationships. In the subsequent sections, relationships between hypothesized requirements are traced to refine the understanding of the system.

3.1 Structure

Deconstructing the YDWG-02 revealed its internal structure, from which instances are developed to model the specific hardware and software components. Appendix A.7 represents the *real-world* hardware artifacts reconstructed and modeled in SysML. Following SysML best practices, structural elements are traced in a separate requirements diagram shown in Appendix A.8 using the satisfy relationship.

3.2 Behavior

Moving toward behavioral understanding, activity and use case diagrams are used to model *real-world* understanding of the system's features. Traces, such as the wireshark capture in Appendix A.9, increase the confidence of features described in the user manual (i.e., NMEA 0183 over TCP/IP). Again, following SysML best practices, behavioral elements are traced in a separate requirements diagram shown in Appendix A.10 using the refine relationship.

3.3 Threats

After ending the initial reverse engineering efforts, the original requirements developed are traced to potential threats found in the system some of which are shown in Appendix A.11 using the potential threat relationship. These potential threats are weaknesses, described by MITRE, as the root cause; otherwise, what leads to vulnerabilities. Those weaknesses are then evaluated based on risk. MITRE provides the CWSS score to analyze importance, exploitability, and environmental impact; however, SysML provides traceability matrices that provide the same value (see Appendix A.12). The following CWEs were found to have more hits on environmental impacts (i.e., more requirements covered): CWE-1428: Reliance on HTTP instead of HTTPS, CWE-1391: Use of Weak Credentials, and CWE-306: Missing Authentication for Critical Function. After classifying these weaknesses as critical failure points in the security of the YDWG-02, further evaluation performed on the device proved that each of these threat interfaces (i.e., CWEs) confirmed to be vulnerabilities and could be exploited to remotely control the operation of a marine vessel (see Appendix A.13, and A.14) [23]. What's important is that each CWE is refined by a vulnerability and then attack pattern, following the causal chain *Weakness* → *Vulnerability* → *Attack Pattern* described by MITRE. The traceability matrix in Appendix A.16 is then served to track and visualize these threats and eventually derive mitigations, using threaten and mitigate stereotypes between requirements and weaknesses.

3.4 Mitigations

Using the threats realized in the traceability matrix, potential mitigations are hypothesized and traced in Appendix A.17 using the potential mitigation relationship. George Santayana famously

quotes, "Those who cannot remember the past are condemned to repeat it." For this case study, the National Institute of Standards and Technology (NIST) Special Publication (SP) 800-82r3 Guide to Operational Technology (OT) Security was added to the CVMS library to provide the most actionable guidance and best practice, backed by security research and vulnerabilities found throughout history [24]. From the NIST library a few cybersecurity requirements are derived from guidance 6.2.10 (remote access) and 6.2.1.3 (network segmentation and isolation): "The marine gateway shall enforce TLSv1.2 at the least, at all times under operation", "The marine gateway shall enforce strong passwords with at least 8 characters, 1 uppercase character, 1 lowercase character, 1 digit, and 1 special character, at all times under operation", and "The marine gateway shall default to blacklisting any NMEA 2000 messages sent on the NMEA bus during initial setup," to cover the higher risk threats highlighted earlier by the CWEs.

3.4.1 Implementation

The implemented system operates as a secure reverse proxy that places an NGINX HTTPS front-end in front of a Python-based smart proxy. All client traffic is first terminated by NGINX over TLS, ensuring encrypted transport, and is then forwarded to the Python application running locally. This allows the Python layer to apply security policies and inspect requests before any communication reaches the underlying legacy device (YDWG-02) [23].

NGINX itself performs only TLS termination and request forwarding. It redirects all HTTP traffic to HTTPS, loads the appropriate server certificates, normalizes incoming headers, and sends every request to the Python proxy. No authentication or device-specific logic is implemented at this layer; it serves purely as a secure gateway.

The Python smart proxy examines and filters each request before deciding whether it should be forwarded to the YDWG-02. It enforces a strong password policy for both login attempts and password changes, requiring a minimum length and specific combinations of character types. Attempts to use weak passwords are intercepted immediately and blocked, preventing them from reaching the YDWG-02.

To prevent the system from operating under factory-default or insecure credentials, the proxy maintains an internal state indicating whether a secure administrative password has been set. While the YDWG-02 remains in this insecure state, only a minimal set of safe endpoints is allowed. All other requests are redirected to the administrative interface until a compliant password is established.

For allowed requests, the proxy reconstructs the backend URL, forwards the HTTP method, body, and cookies, and removes headers that should not be passed downstream. Responses from the YDWG-02 are returned directly to the client with cleaned headers, making the interaction appear transparent once the YDWG-02 has been secured.

Upon startup, the proxy issues an initialization request intended to reset the default firewall filters. This ensures the YDWG-02 begins in a known and stable state (no transmission of NMEA networking control logic allowed).

Overall, the architecture combines strong TLS termination with programmable request interception, enabling secure operation of a legacy web interface without requiring any modifications to the YDWG-02 itself.

3.4.2 Testing

Testing the implementation, a suite of pytests were developed [23]. An example of the supporting code is shown in Appendix B.1 and it's interfaced with SysML using the pytest activity seen in Appendix A.19. This allows for reusability, and verification of the cybersecurity requirements, shown in Appendix A.20 and A.18.

3.4.3 Vulnerabilities

Access to the proxy server (Nginx + Python3) can diminish trust in the system, as threats may be able to modify traffic or sniff sensitive information. Any compromised administrator account could expose cached cookies, opening the door to additional web-based exploits such as XSS or clickjacking. Because the firmware isn't encrypted, a motivated actor could extract secrets directly from the device and potentially reassemble or even update it with a malicious version. However, using SysML these potential threats can be iteratively refined based on incentives or risk factors.

4 Reflection

Anderson's work shaped my project by reframing the YDWG-02 Marine Gateway not merely as a device but as a collection of interacting protocols such as Wi-Fi, HTTP, NMEA-2000, and vendor-specific flows, each capable of failing through incorrect assumptions about message exchange, trust boundaries, or context. His discussions of protocol mistakes, reflection attacks, weak defaults, and cryptographic misuse clarified why threats such as reliance on HTTP instead of HTTPS (CWE-1428 to CAPEC-94) and command-execution paths lacking proper authentication (CWE-306 to CAPEC-248) emerge so easily in embedded systems. Chapters 5 and 6 further influenced the project by highlighting how weak key handling, unsafe defaults, persistent default credentials, and poorly engineered access boundaries lead systems to trust encrypted equals as secure or treat authentication as an afterthought. These issues directly reflected in the YDWG-02 plain-text management interface and vulnerabilities such as CWE-1391 and CWE-306. These insights informed mitigations like CSR 1.2 for strong password enforcement and default-credential removal, as demonstrated in the SysML implementations. Finally, Anderson's observations about inconsistent tamper resistance in commercial IoT devices helped justify why physical extraction attacks remain feasible and why network-layer defenses alone cannot secure a marine IoT gateway operating in a hostile mixed IT and OT environment.

References

- [1] OMG systems modeling language (OMG SysML), December 2019. tex.entrytype: standard.
- [2] M. Ataiefard and others. Deep state inference for cyber-physical systems via hybrid neural networks. *arXiv preprint*, 2021.
- [3] United States National Security Agency (NSA). Ghidra: Open source software reverse engineering framework, 2019.
- [4] Martin Hochwallner and others. Some Aspects of SysML Application in the Reverse Engineering of Mechatronic Systems. In *Mechatronic Systems and Materials*, pages 115–126. Springer, 2011.
- [5] Robert Karban and others. Exploring Model Based Engineering for Large Telescopes – Getting started with descriptive models. In *SPIE Conference on Software and Cyberinfrastructure for Astronomy*, 2016.
- [6] Andreas Sailer. *Reverse Engineering of Real-Time System Models from Event Trace Recordings*. PhD Thesis, University of Bamberg, 2021.
- [7] Open-CISO. Understand common attack patterns, March 2024. Authority: fnCyber, Inc.
- [8] Mitre, 1958. Place: Bedford, Massachusetts and McLean, Virginia, USA tex.entrytype: organization.
- [9] National Marine Electronics Association (NMEA). NMEA0183-2, January 2002.
- [10] National Marine Electronics Association (NMEA). Nmea 2000® — the modern marine interface standard: Can-based data communications for vessel systems integration, 2022. Version 3.000 (with amendments) as published by NMEA.
- [11] National Marine Electronics Association (NMEA). OneNet – the standard for IP networking of marine electronic devices, 2023. Place: Annapolis, MD, USA tex.entrytype: standard.
- [12] CitiMarine Store. What is NMEA 2000? The smart backbone of modern marine electronics, June 2025.
- [13] National Marine Electronics Association (NMEA). Best marine electronic gateways, November 2024.
- [14] Yacht Devices Ltd. NMEA 0183 wi-fi gateway YDWN-02, April 2018.
- [15] Maretron LLC. J2K100 J1939 to NMEA 2000®gateway, 2025.
- [16] Marko Vukšić, Jasmin Ćelić, Ivan Panić, and Aleksandar Cuculić. Exploiting maritime wi-fi: Practical assessment of onboard network vulnerabilities. *Journal of Marine Science and Engineering*, 13(8):1576, 2025.
- [17] Visure Solutions. The V model in systems engineering, 2024.
- [18] Yacht Devices Ltd. NMEA 2000 wi-fi gateway YDWG-02, 2025.
- [19] Yacht Devices Ltd. News, 2025.

- [20] curtdo. Latest on NMEA to WiFi?, August 2023. Authority: SB Owners Forums.
- [21] Fredrik Ahlberg, Angus Gratton, and Espressif Systems. esptool.py: Serial utility for flashing, provisioning, and interacting with Espressif SoCs, 2025.
- [22] John Matherly. Shodan: The internet of things search engine, 2009.
- [23] Spenc3rB. yachtDestroy, 2025.
- [24] Keith A. Stouffer, Michael Pease, CheeYee Tang, Timothy Zimmerman, Victoria Yan Pillitteri, Suzanne Lightman, Adam Hahn, Stephanie Saravia, Aslam Sherule, and Michael Thompson. Guide to operational technology (OT) security. Special Publication 800-82 Revision 3 SP 800-82 r3, National Institute of Standards and Technology (NIST), September 2023.

A Supporting SysML Figures

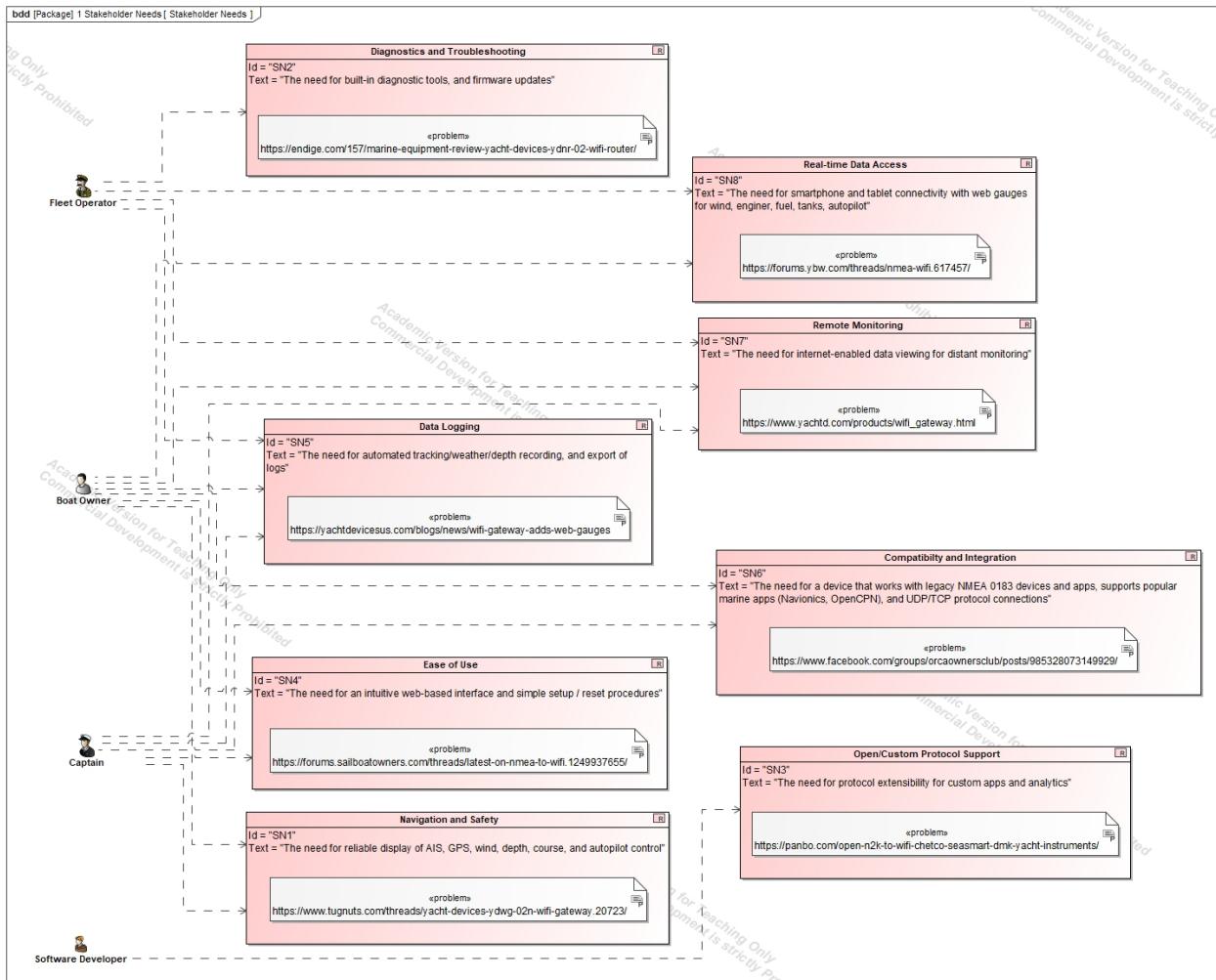


Figure A.1: Stakeholder Needs Diagram Derived from Reconnaissance

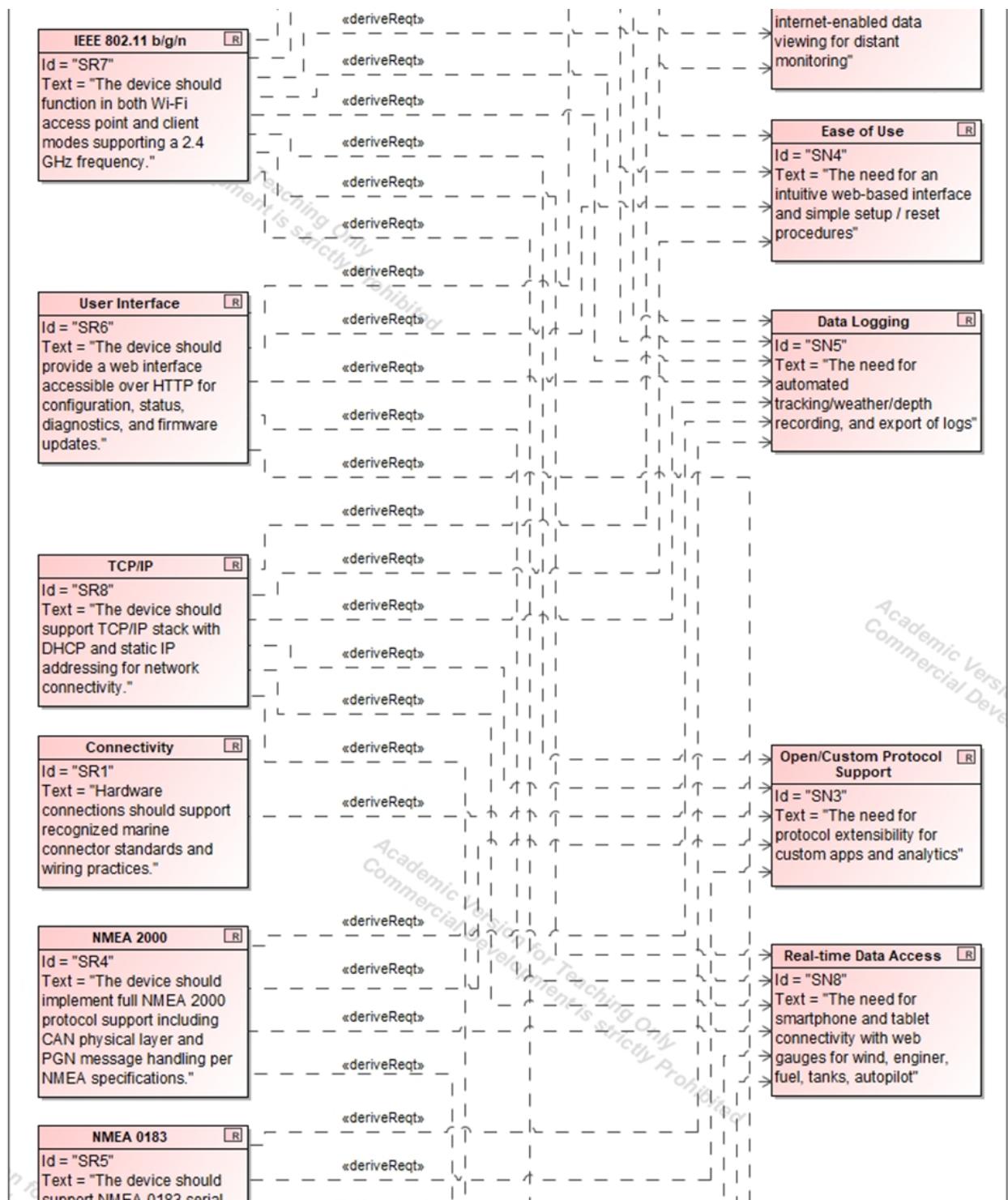


Figure A.2: Inferred System Requirements Diagram Snippet

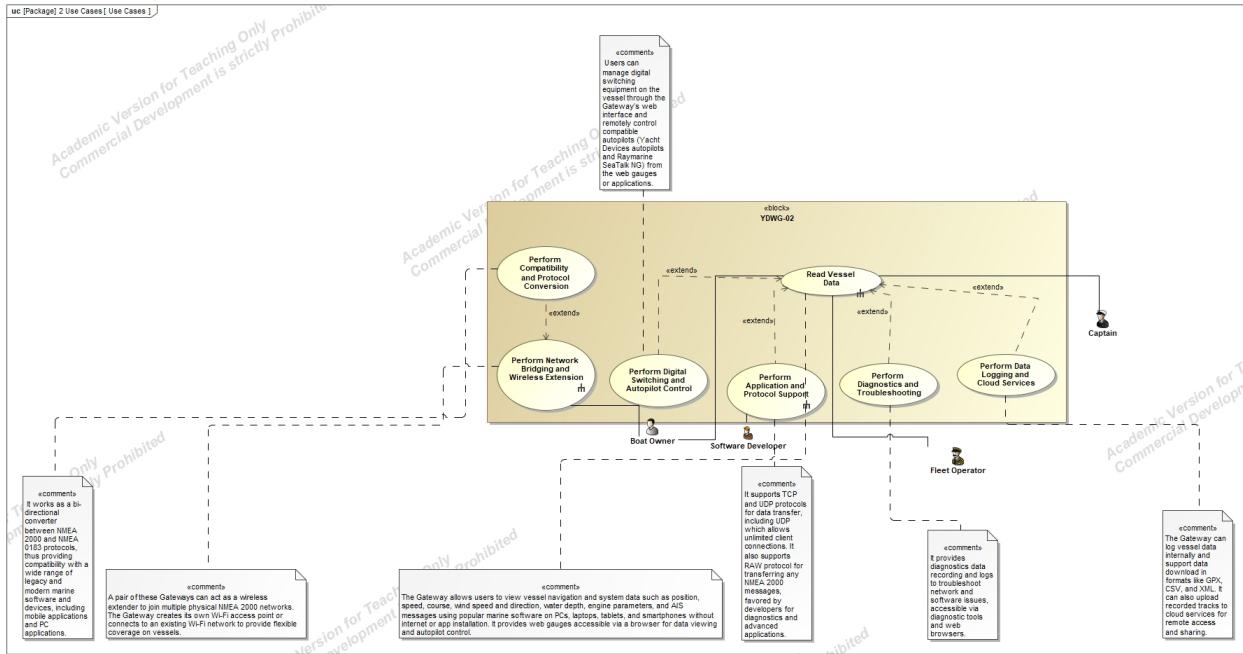


Figure A.3: Behavior Diagram Derived from Reconnaissance

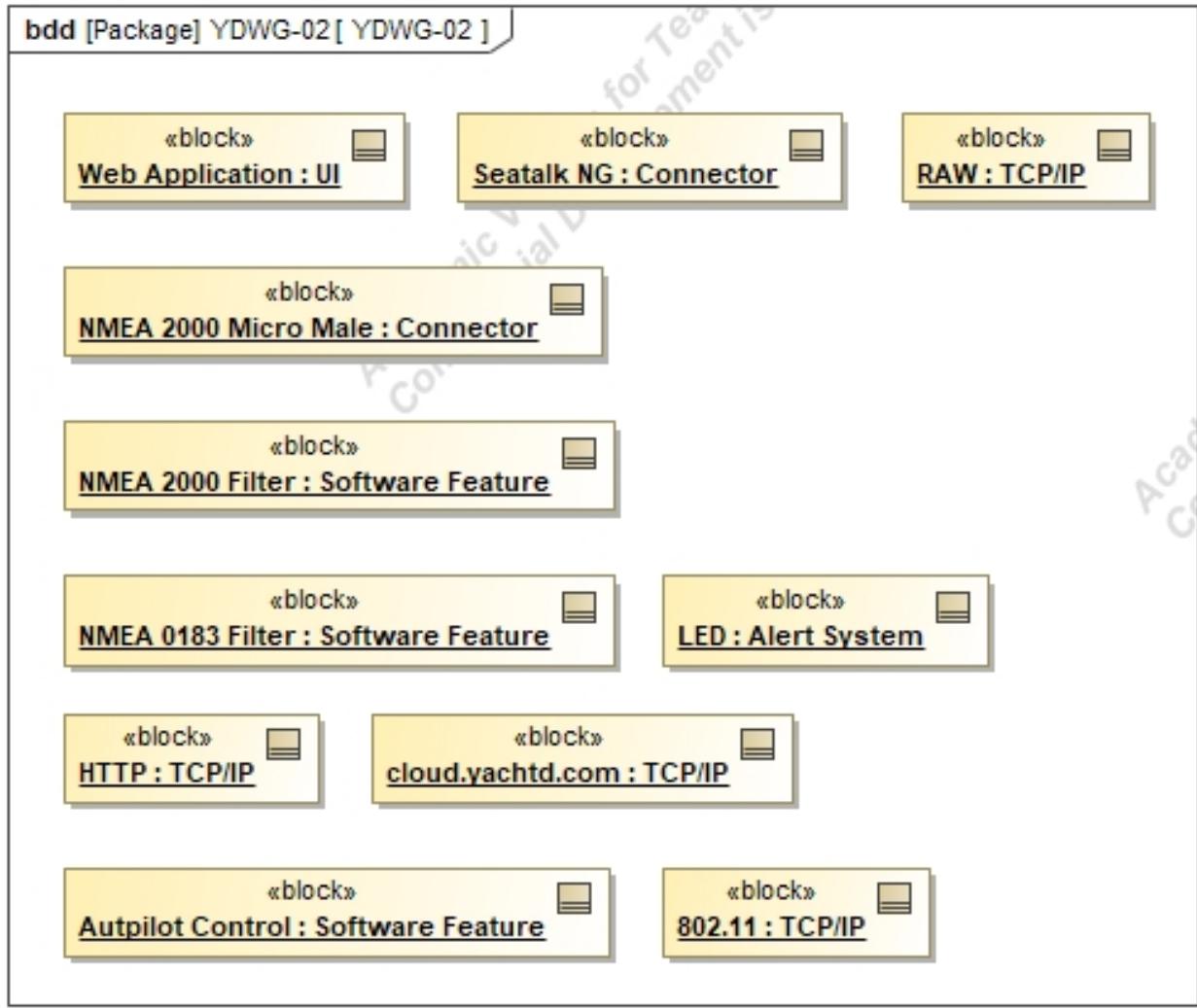


Figure A.4: Structure Diagram Derived from Reconnaissance

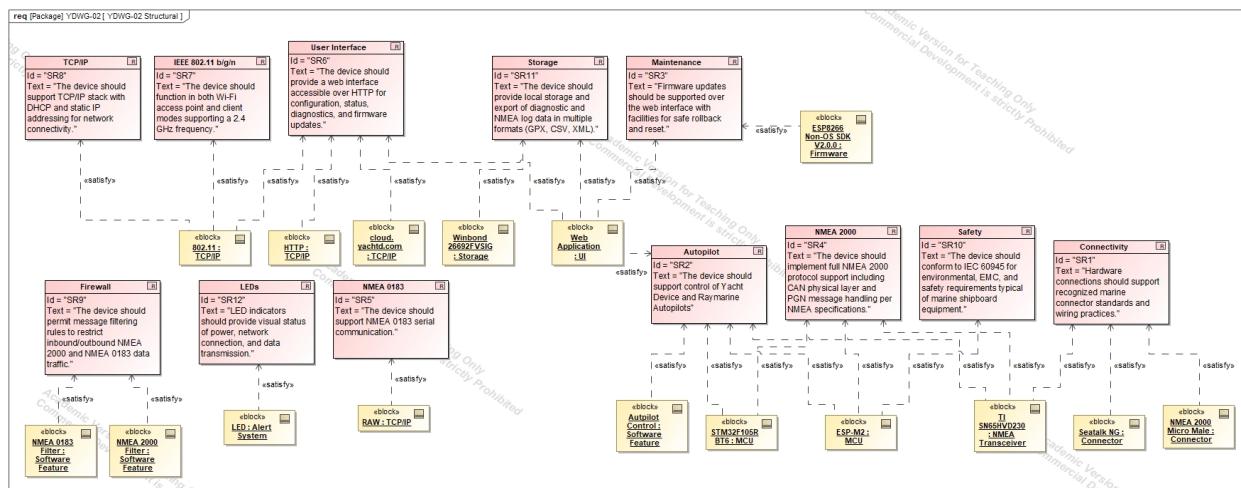


Figure A.5: Structure Traced from Reverse Engineering

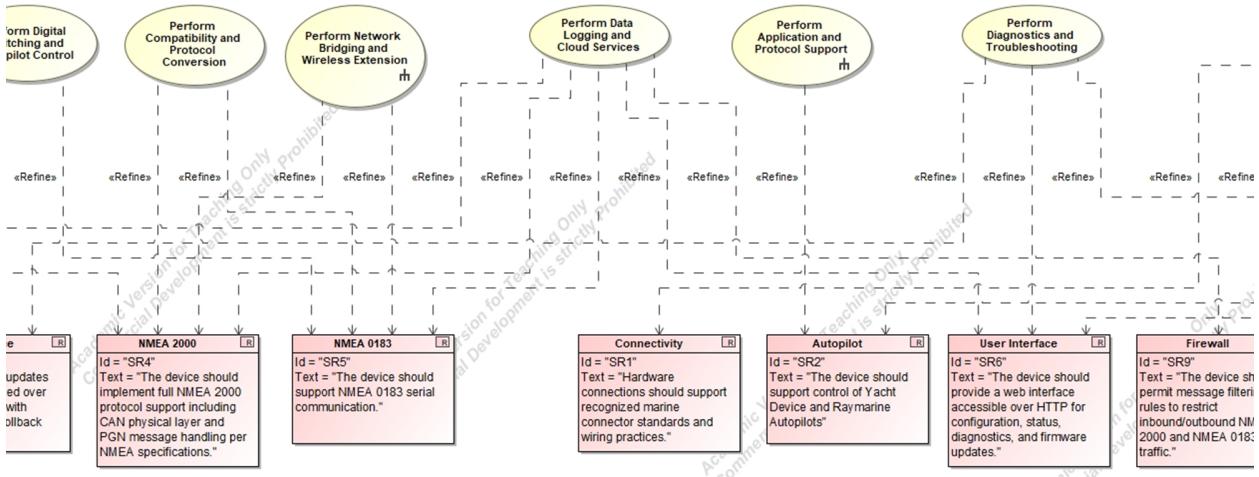


Figure A.6: Behavior Traced from Reverse Engineering

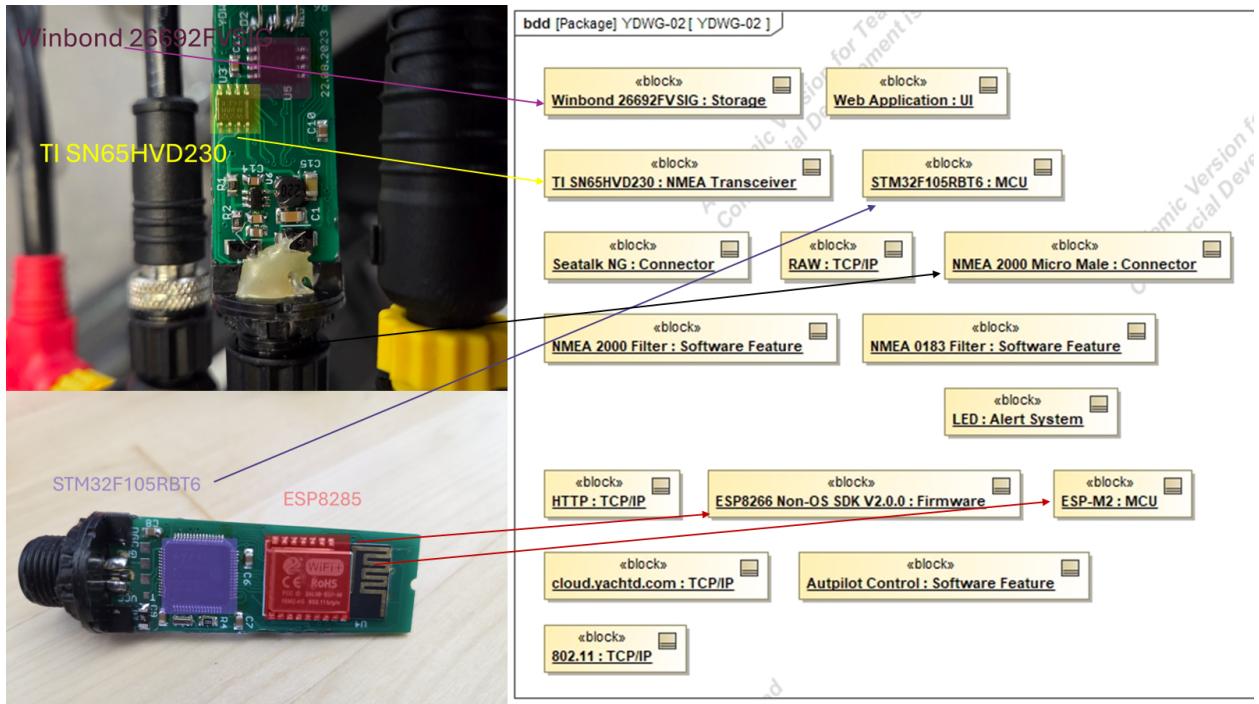


Figure A.7: YDWG-02 Structural Findings

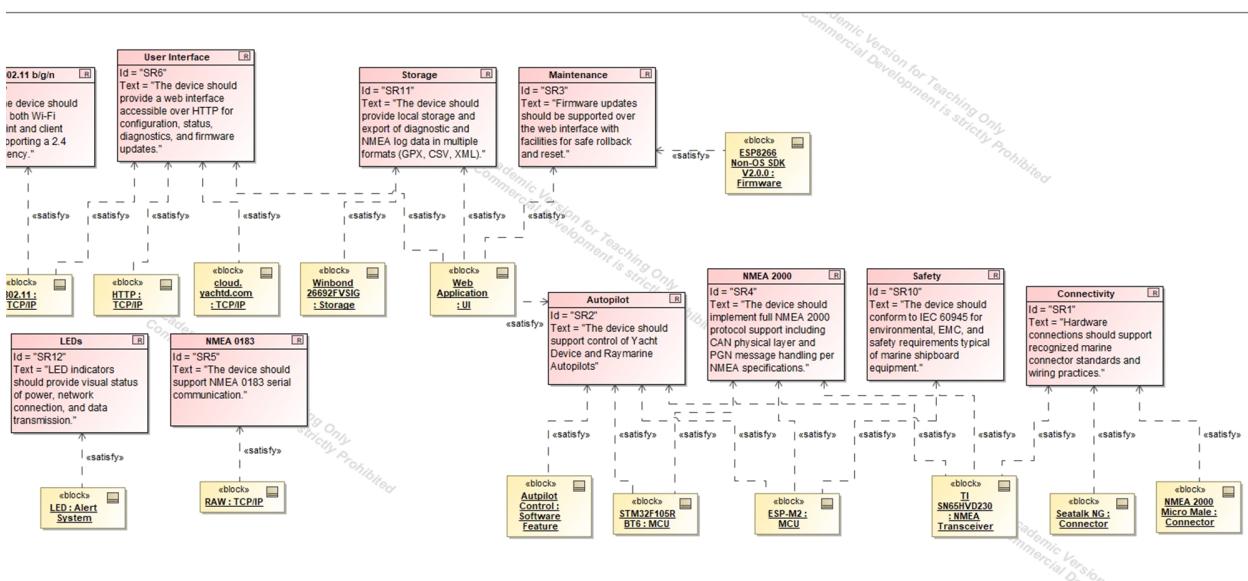


Figure A.8: YDWG-02 Structural Relationships

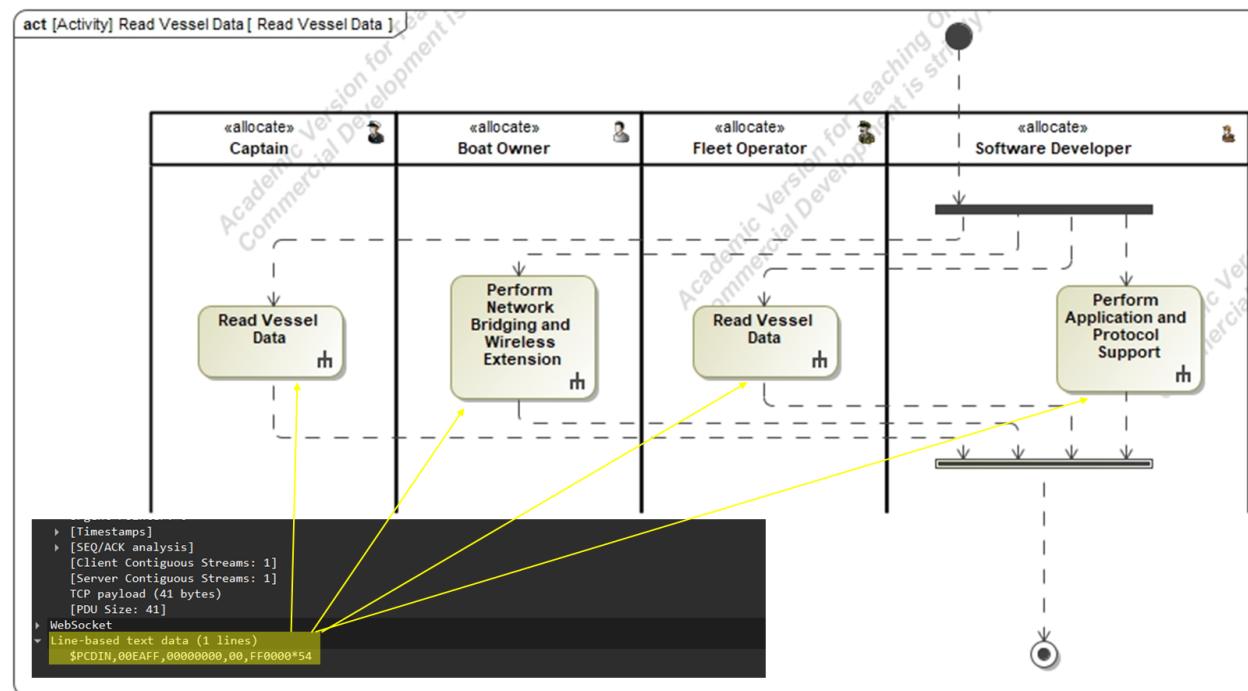


Figure A.9: YDWG-02 Behavioral Findings

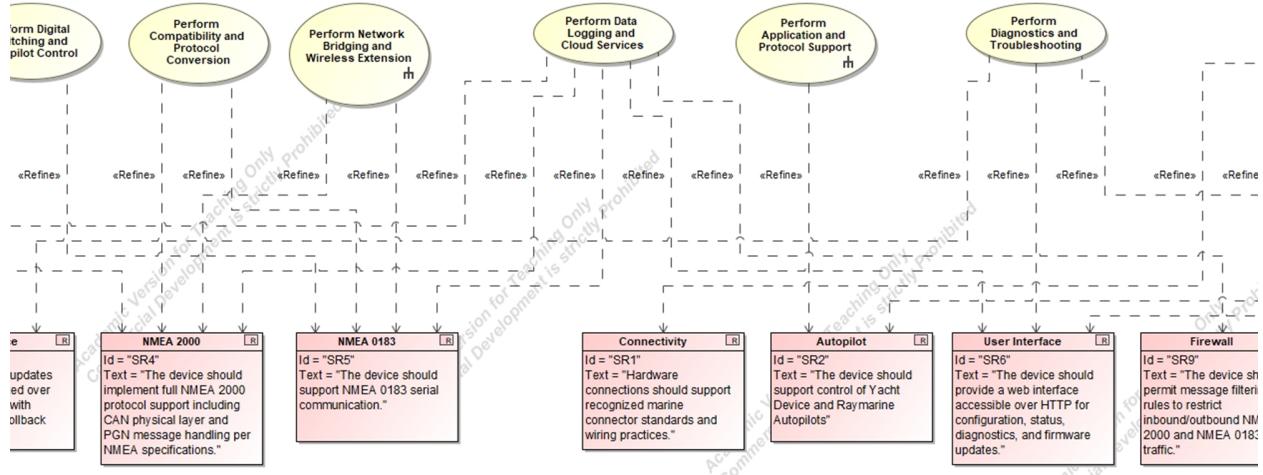


Figure A.10: YDWG-02 Behavioral Relationships

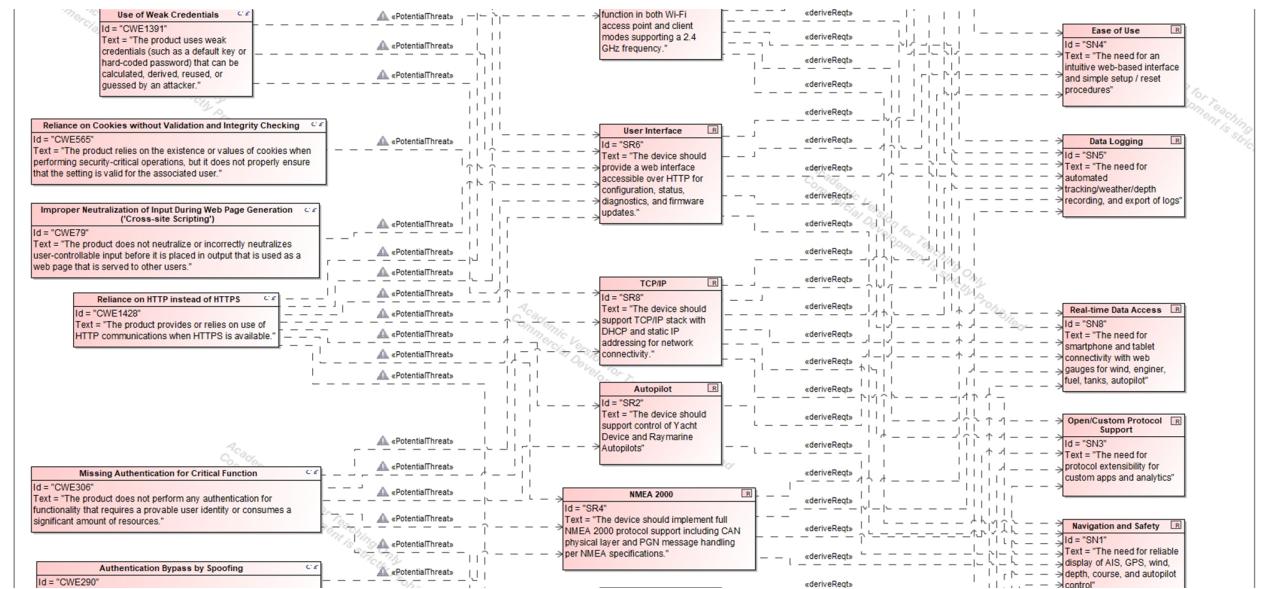


Figure A.11: Potential Threats (Weaknesses)

*Academic Version for Teaching Only
Commercial Development is strictly Prohibited*

		Marine Gateways Solution Domain																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
		SR1	SR2	SR3	SR4	SR5	SR6	SR7	SR8	SR9	SR10	SR11	SR12	SR13	SR14	SR15	SR16	SR17	SR18	SR19	SR20	SR21	SR22	SR23	SR24	SR25	SR26	SR27	SR28	SR29	SR30	SR31	SR32	SR33	SR34	SR35	SR36	SR37	SR38	SR39	SR40	SR41	SR42	SR43	SR44	SR45	SR46	SR47	SR48	SR49	SR50	SR51	SR52	SR53	SR54	SR55	SR56	SR57	SR58	SR59	SR60	SR61	SR62	SR63	SR64	SR65	SR66	SR67	SR68	SR69	SR70	SR71	SR72	SR73	SR74	SR75	SR76	SR77	SR78	SR79	SR80	SR81	SR82	SR83	SR84	SR85	SR86	SR87	SR88	SR89	SR90	SR91	SR92	SR93	SR94	SR95	SR96	SR97	SR98	SR99	SR100	SR101	SR102	SR103	SR104	SR105	SR106	SR107	SR108	SR109	SR110	SR111	SR112	SR113	SR114	SR115	SR116	SR117	SR118	SR119	SR120	SR121	SR122	SR123	SR124	SR125	SR126	SR127	SR128	SR129	SR130	SR131	SR132	SR133	SR134	SR135	SR136	SR137	SR138	SR139	SR140	SR141	SR142	SR143	SR144	SR145	SR146	SR147	SR148	SR149	SR150	SR151	SR152	SR153	SR154	SR155	SR156	SR157	SR158	SR159	SR160	SR161	SR162	SR163	SR164	SR165	SR166	SR167	SR168	SR169	SR170	SR171	SR172	SR173	SR174	SR175	SR176	SR177	SR178	SR179	SR180	SR181	SR182	SR183	SR184	SR185	SR186	SR187	SR188	SR189	SR190	SR191	SR192	SR193	SR194	SR195	SR196	SR197	SR198	SR199	SR200	SR201	SR202	SR203	SR204	SR205	SR206	SR207	SR208	SR209	SR210	SR211	SR212	SR213	SR214	SR215	SR216	SR217	SR218	SR219	SR220	SR221	SR222	SR223	SR224	SR225	SR226	SR227	SR228	SR229	SR230	SR231	SR232	SR233	SR234	SR235	SR236	SR237	SR238	SR239	SR240	SR241	SR242	SR243	SR244	SR245	SR246	SR247	SR248	SR249	SR250	SR251	SR252	SR253	SR254	SR255	SR256	SR257	SR258	SR259	SR260	SR261	SR262	SR263	SR264	SR265	SR266	SR267	SR268	SR269	SR270	SR271	SR272	SR273	SR274	SR275	SR276	SR277	SR278	SR279	SR280	SR281	SR282	SR283	SR284	SR285	SR286	SR287	SR288	SR289	SR290	SR291	SR292	SR293	SR294	SR295	SR296	SR297	SR298	SR299	SR300	SR301	SR302	SR303	SR304	SR305	SR306	SR307	SR308	SR309	SR310	SR311	SR312	SR313	SR314	SR315	SR316	SR317	SR318	SR319	SR320	SR321	SR322	SR323	SR324	SR325	SR326	SR327	SR328	SR329	SR330	SR331	SR332	SR333	SR334	SR335	SR336	SR337	SR338	SR339	SR340	SR341	SR342	SR343	SR344	SR345	SR346	SR347	SR348	SR349	SR350	SR351	SR352	SR353	SR354	SR355	SR356	SR357	SR358	SR359	SR360	SR361	SR362	SR363	SR364	SR365	SR366	SR367	SR368	SR369	SR370	SR371	SR372	SR373	SR374	SR375	SR376	SR377	SR378	SR379	SR380	SR381	SR382	SR383	SR384	SR385	SR386	SR387	SR388	SR389	SR390	SR391	SR392	SR393	SR394	SR395	SR396	SR397	SR398	SR399	SR400	SR401	SR402	SR403	SR404	SR405	SR406	SR407	SR408	SR409	SR410	SR411	SR412	SR413	SR414	SR415	SR416	SR417	SR418	SR419	SR420	SR421	SR422	SR423	SR424	SR425	SR426	SR427	SR428	SR429	SR430	SR431	SR432	SR433	SR434	SR435	SR436	SR437	SR438	SR439	SR440	SR441	SR442	SR443	SR444	SR445	SR446	SR447	SR448	SR449	SR450	SR451	SR452	SR453	SR454	SR455	SR456	SR457	SR458	SR459	SR460	SR461	SR462	SR463	SR464	SR465	SR466	SR467	SR468	SR469	SR470	SR471	SR472	SR473	SR474	SR475	SR476	SR477	SR478	SR479	SR480	SR481	SR482	SR483	SR484	SR485	SR486	SR487	SR488	SR489	SR490	SR491	SR492	SR493	SR494	SR495	SR496	SR497	SR498	SR499	SR500	SR501	SR502	SR503	SR504	SR505	SR506	SR507	SR508	SR509	SR510	SR511	SR512	SR513	SR514	SR515	SR516	SR517	SR518	SR519	SR520	SR521	SR522	SR523	SR524	SR525	SR526	SR527	SR528	SR529	SR530	SR531	SR532	SR533	SR534	SR535	SR536	SR537	SR538	SR539	SR540	SR541	SR542	SR543	SR544	SR545	SR546	SR547	SR548	SR549	SR550	SR551	SR552	SR553	SR554	SR555	SR556	SR557	SR558	SR559	SR560	SR561	SR562	SR563	SR564	SR565	SR566	SR567	SR568	SR569	SR570	SR571	SR572	SR573	SR574	SR575	SR576	SR577	SR578	SR579	SR580	SR581	SR582	SR583	SR584	SR585	SR586	SR587	SR588	SR589	SR590	SR591	SR592	SR593	SR594	SR595	SR596	SR597	SR598	SR599	SR600	SR601	SR602	SR603	SR604	SR605	SR606	SR607	SR608	SR609	SR610	SR611	SR612	SR613	SR614	SR615	SR616	SR617	SR618	SR619	SR620	SR621	SR622	SR623	SR624	SR625	SR626	SR627	SR628	SR629	SR630	SR631	SR632	SR633	SR634	SR635	SR636	SR637	SR638	SR639	SR640	SR641	SR642	SR643	SR644	SR645	SR646	SR647	SR648	SR649	SR650	SR651	SR652	SR653	SR654	SR655	SR656	SR657	SR658	SR659	SR660	SR661	SR662	SR663	SR664	SR665	SR666	SR667	SR668	SR669	SR670	SR671	SR672	SR673	SR674	SR675	SR676	SR677	SR678	SR679	SR680	SR681	SR682	SR683	SR684	SR685	SR686	SR687	SR688	SR689	SR690	SR691	SR692	SR693	SR694	SR695	SR696	SR697	SR698	SR699	SR700	SR701	SR702	SR703	SR704	SR705	SR706	SR707	SR708	SR709	SR710	SR711	SR712	SR713	SR714	SR715	SR716	SR717	SR718	SR719	SR720	SR721	SR722	SR723	SR724	SR725	SR726	SR727	SR728	SR729	SR730	SR731	SR732	SR733	SR734	SR735	SR736	SR737	SR738	SR739	SR740	SR741	SR742	SR743	SR744	SR745	SR746	SR747	SR748	SR749	SR750	SR751	SR752	SR753	SR754	SR755	SR756	SR757	SR758	SR759	SR760	SR761	SR762	SR763	SR764	SR765	SR766	SR767	SR768	SR769	SR770	SR771	SR772	SR773	SR774	SR775	SR776	SR777	SR778	SR779	SR780	SR781	SR782	SR783	SR784	SR785	SR786	SR787	SR788	SR789	SR790	SR791	SR792	SR793	SR794	SR795	SR796	SR797	SR798	SR799	SR800	SR801	SR802	SR803	SR804	SR805	SR806	SR807	SR808	SR809	SR8010	SR8011	SR8012	SR8013	SR8014	SR8015	SR8016	SR8017	SR8018	SR8019	SR8020	SR8021	SR8022	SR8023	SR8024	SR8025	SR8026	SR8027	SR8028	SR8029	SR8030	SR8031	SR8032	SR8033	SR8034	SR8035	SR8036	SR8037	SR8038	SR8039	SR8040	SR8041	SR8042	SR8043	SR8044	SR8045	SR8046	SR8047	SR8048	SR8049	SR8050	SR8051	SR8052	SR8053	SR8054	SR8055	SR8056	SR8057	SR8058	SR8059	SR8060	SR8061	SR8062	SR8063	SR8064	SR8065	SR8066	SR8067	SR8068	SR8069	SR8070	SR8071	SR8072	SR8073	SR8074	SR8075	SR8076	SR8077	SR8078	SR8079	SR8080	SR8081	SR8082	SR8083	SR8084	SR8085	SR8086	SR8087	SR8088	SR8089	SR8090	SR8091	SR8092	SR8093	SR8094	SR8095	SR8096	SR8097	SR8098	SR8099	SR80100	SR80101	SR80102	SR80103	SR80104	SR80105	SR80106	SR80107	SR80108	SR80109	SR80110	SR80111	SR80112	SR80113	SR80114	SR80115	SR80116	SR80117	SR80118	SR80119	SR80120	SR80121	SR80122	SR80123	SR80124	SR80125	SR80126	SR80127	SR80128	SR80129	SR80130	SR80131	SR80132	SR80133	SR80134	SR80135	SR80136	SR80137	SR80138	SR80139	SR80140	SR80141	SR80142	SR80143	SR80144	SR80145	SR80146	SR80147	SR80148	SR80149	SR80150	SR80151	SR80152	SR80153	SR80154	SR80155	SR80156	SR80157	SR80158	SR80159	SR80160	SR80161	SR80162	SR80163	SR80164	SR80165	SR80166	SR80167	SR80168	SR80169	SR80170	SR80171	SR80172	SR80173	SR80174	SR80175	SR80176	SR80177	SR80178	SR80179	SR80180	SR80181	SR80182	SR80183	SR80184	SR80185	SR80186	SR80187	SR80188	SR80189	SR80190	SR80191	SR80192	SR80193	SR80194	SR80195	SR80196	SR80197	SR80198	SR80199	SR80200	SR80201	SR80202	SR80203	SR80204	SR80205	SR80206	SR80207	SR80208	SR80209	SR80210	SR80211	SR80212	SR80213	SR80214	SR80215	SR80216	SR80217	SR80218	SR80219	SR80220	SR80221	SR80222	SR80223	SR80224	SR80225	SR80226	SR80227	SR80228	SR80229	SR80230	SR80231	SR80232	SR80233	SR80234	SR80235	SR80236	SR80237	SR80238	SR80239	SR80240	SR80241	SR80242	SR80243	SR80244	SR80245	SR80246	SR80247	SR80248	SR80249	SR80250	SR80251	SR80252	SR80253	SR80254	SR80255	SR80256	SR80257	SR80258	SR80259	SR80260	SR80261	SR80262	SR80263	SR80264	SR80265	SR80266	SR80267	SR80268	SR80269	SR80270	SR80271	SR80272	SR80273	SR80274	SR80275	SR80276	SR80277	SR80278	SR80279	SR80280	SR80281	SR80282	SR80283	SR80284	SR80285	SR80286	SR80287	SR80288	SR80289	SR80290	SR80291	SR80292	SR80293	SR80294	SR80295	SR80296	SR80297	SR80298	SR80299	SR80300	SR80301	SR80302	SR80303	SR80304	SR80305	SR80306	SR80307	SR80308	SR80309	SR80310	SR80311	SR80312	SR80313	SR80314	SR80315	SR80316	SR80317	SR80318	SR80319	SR80320	SR80321	SR80322	SR80323	SR80324	SR80325	SR80326	SR80327	SR80328	SR80329	SR80330	SR80331	SR80332	SR80333	SR80334	SR80335	SR80336	SR80337	SR80338	SR80339	SR80340	SR80341	SR80342	SR80343	SR80344	SR80345	SR80346	SR80347	SR80348	SR80349	SR80350	SR80351	SR80352	SR80353	SR80354	SR80355	SR80356	SR80357	SR80358	SR80359	SR80360	SR80361	SR80362	SR80363	SR80364	SR80365	SR80366	SR80367	SR80368	SR80369	SR80370	SR80371	SR80372	SR80373	SR80374	SR80375	SR80376	SR80377	SR80378	SR80379	SR80380	SR80381	SR80382	SR80383	SR80384	SR80385	SR80386	SR80387	SR80388	SR80389	SR80390	SR80391	SR80392	SR8

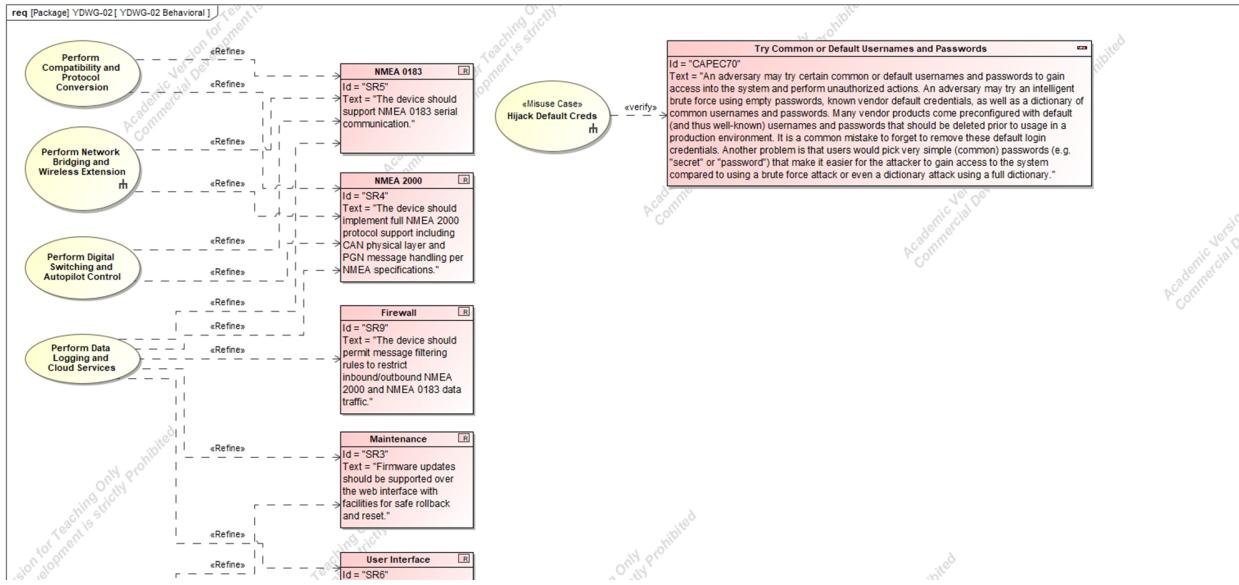


Figure A.14: YDWG-02 Attack Pattern Traceability

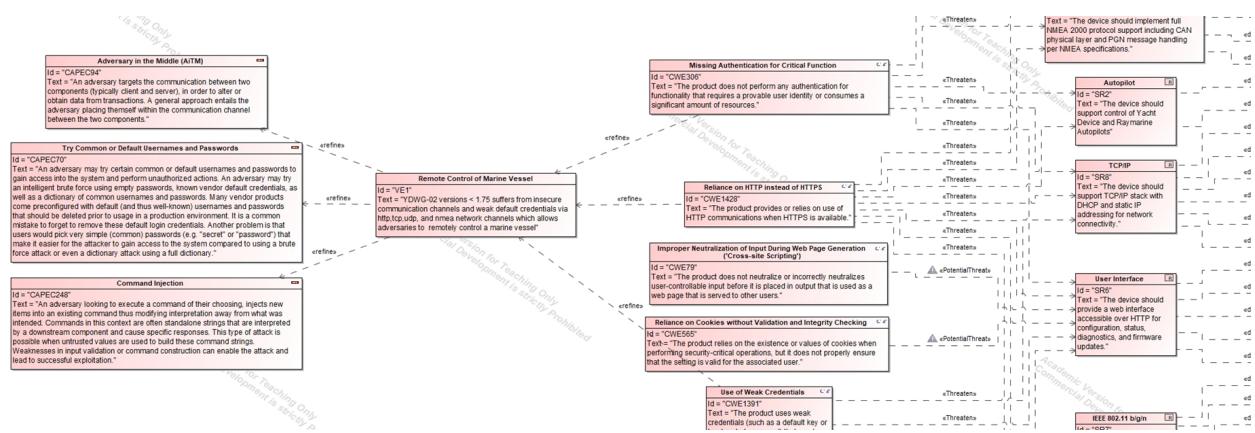


Figure A.15: YDWG-02 Requirements Refinement via Cause, Vulnerability, and Exploit

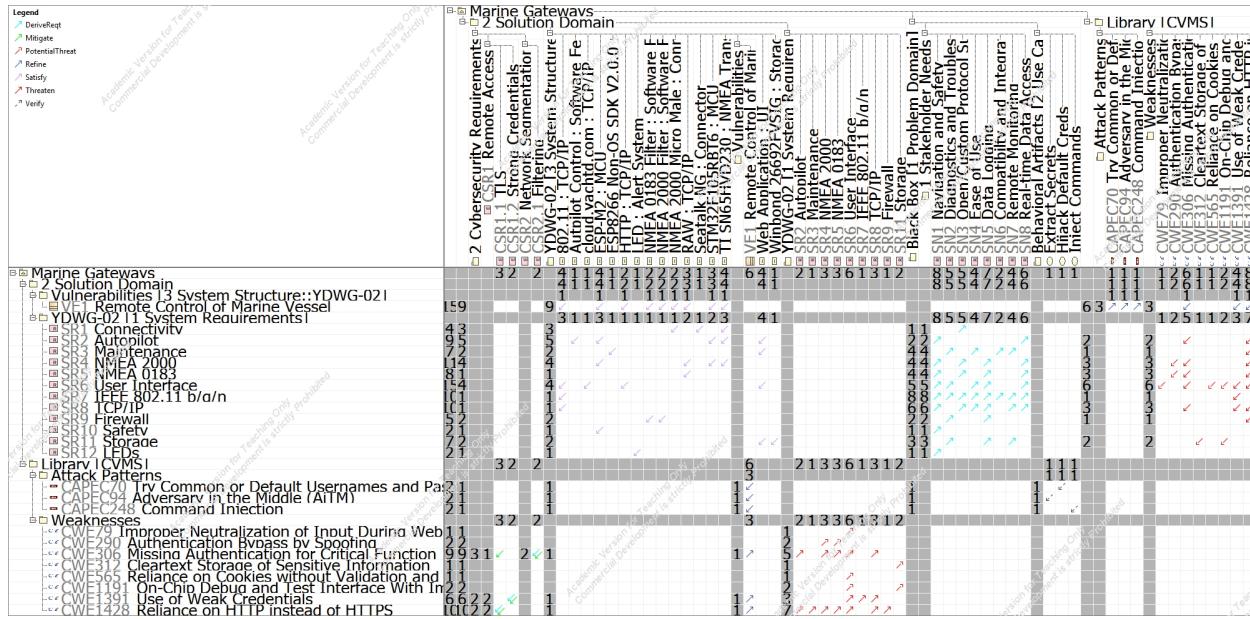


Figure A.16: Full Traceability Matrix

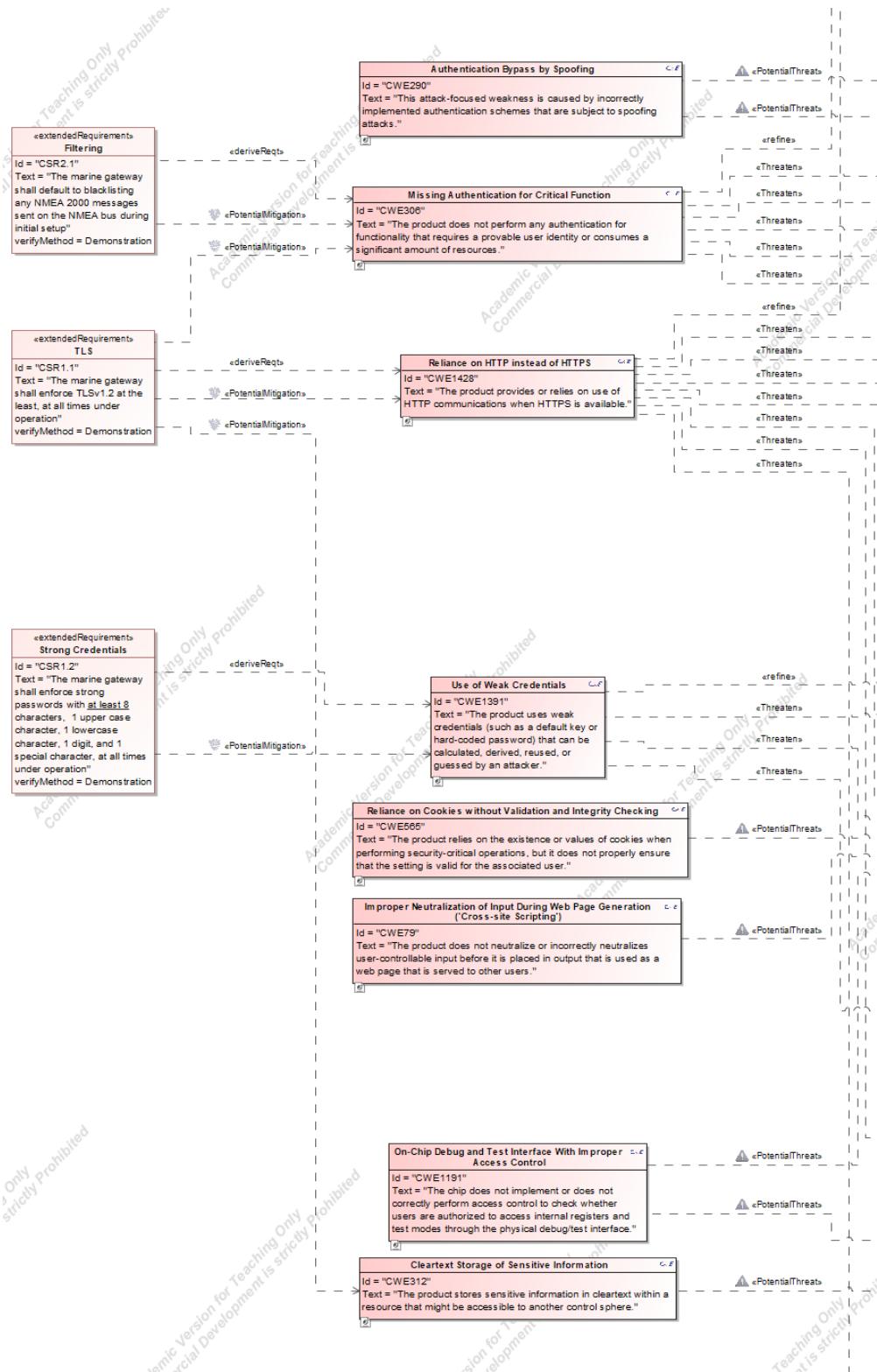


Figure A.17: YDWG-02 Potential Mitigations

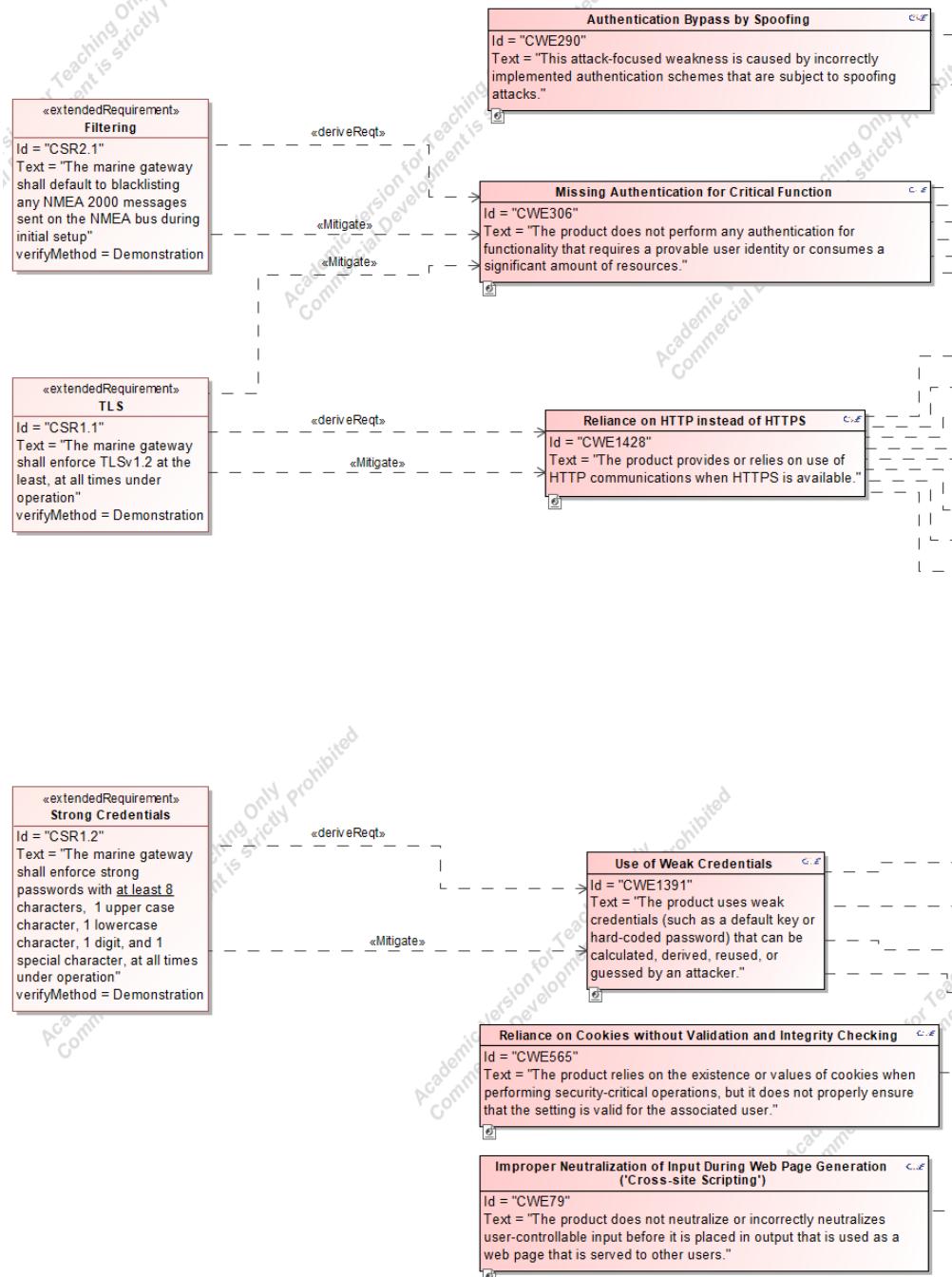


Figure A.18: YDWG-02 Mitigations

#	Name	Text	Satisfied By	Derived From	Verified By	Verify Method
1	CSR1 Remote Access	The marine gateway shall enforce TLSv1.2 at the least, at all times under operation	Nginx+Python3 Smart Proxy	CWE1428 Reliance on HTTP instead of HTTPS	pytest	Inspection
2	CSR1.1 TLS	The marine gateway shall enforce strong passwords with at least 8 characters, 1 uppercase character, 1 lowercase character, 1 digit, and 1 special character, at all times under operation	Nginx+Python3 Smart Proxy	CWE1391 Use of Weak Credentials	pytest	Demonstration
3	CSR1.2 Strong Credentials		Nginx+Python3 Smart Proxy			Demonstration
4	CSR2 Network Segmentation and Isolation	The marine gateway shall default to blacklisting any NMEA 2000 messages sent on the NMEA bus during initial setup	Nginx+Python3 Smart Proxy	CWE306 Missing Authentication for Critical Function	pytest	Inspection
5	CSR2.1 Filtering		Nginx+Python3 Smart Proxy			Demonstration

Figure A.20: YDWG-02 Cybersecurity Requirement Table

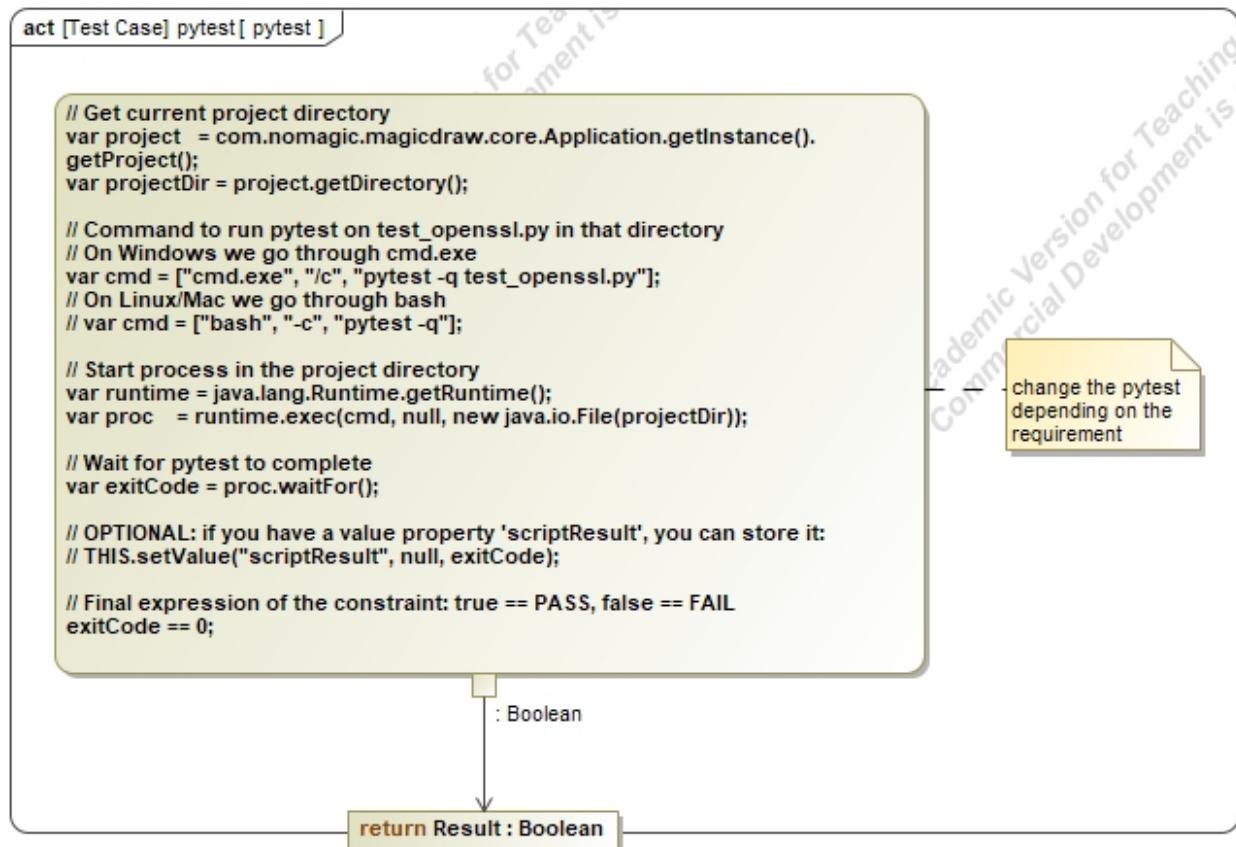


Figure A.19: Unit Testing SysML Medium

B Supporting Code

Algorithm 1: TLS Version Tests Using OpenSSL

```
Function ConnectWithVersion(minv, maxv):
    ctx ← NewTLSContext()
    ctx.setMinProtoVersion(minv)
    ctx.setMaxProtoVersion(maxv)
    sock ← CreateTCPConnection(HOST, PORT)
    conn ← WrapTLS(ctx, sock)
    conn.setConnectState()
    conn.doHandshake()
    return conn.getProtocolVersionName()
    AttemptShutdown(conn)
    Close(sock)
```

Test: TLS 1.3 and TLS 1.2 must succeed

```
negotiated ← ConnectWithVersion (TLS1.3, TLS1.3)
Assert(negotiated = “TLSv1.3”)
negotiated ← ConnectWithVersion (TLS1.2, TLS1.2)
Assert(negotiated = “TLSv1.2”)
```

Test: Old TLS versions must fail

```
foreach (version, name) in {
    (TLS1.1, “TLSv1.1”),  

    (TLS1.0, “TLSv1.0”)
} do
    ExpectException(TLSError)
    ConnectWithVersion (version, version)
```
